Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Информационные системы
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ Л. С. Троценко
подпись
«13» июня 2018 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии

Разработка системы распознавания лиц для пропускной системы института

Руководитель _____ к.т.н., доцент Е.А. Мальцев
подпись, дата

Выпускник _____ Е.А. Никифорова
подпись, дата

Нормоконтролер _____ ст. преподаватель Ю.В. Шмагрис
подпись, дата

Красноярск 2018

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Информационные системы
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ С. А. Виденин
подпись
«2» марта 2018 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту: Никифоровой Екатерины Александровны

Группа: КИ14-13Б Направление: 09.03.02 «Информационные системы и технологии»

Тема выпускной квалификационной работы: Разработка системы распознавания лиц для пропускной системы института.

Утверждена приказом по университету: №4896/с от 05.04.2018 г.

Руководитель ВКР: Е.А. Мальцев, к.т.н., доцент кафедры «Системы искусственного интеллекта» ИКИТ СФУ.

Исходные данные для ВКР: Список требований к разрабатываемой системе, методические указания научного руководителя, учебные пособия.

Перечень разделов ВКР: Обоснование решения о создании ИС, обзор методов обнаружения и распознавания лиц, обзор существующих систем, анализ задачи распознавания лиц в видеопотоках, этапы обработки кадров видеопотока, выбор средств разработки системы, разработка системы распознавания лиц.

Перечень графического материала: Презентация, выполненная в Microsoft Power Point 2016.

Руководитель ВКР _____ Е.А. Мальцев
подпись

Задание принял к исполнению _____ Е.А. Никифорова
подпись

«2» марта 2018 г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка системы распознавания лиц для пропускной системы института» содержит 58 страниц текстового документа, 15 рисунков, 19 использованных источников, 1 приложение.

СИСТЕМА РАСПОЗНАВАНИЯ ЛИЦ, РАСПОЗНАВАНИЕ ОБРАЗОВ, РАСПОЗНАВАНИЕ ЛИЦ, ОБНАРУЖЕНИЕ ЛИЦ, МЕТОД ВИОЛЫ-ДЖОНСА, ЛОКАЛЬНЫЕ БИНАРНЫЕ ШАБЛОНЫ, ЦЕНТРАЛЬНО-СИММЕТРИЧНЫЕ ЛОКАЛЬНЫЕ БИНАРНЫЕ ШАБЛОНЫ.

Объектом разработки является система распознавания лиц в видеопотоках для пропускной системы института.

Целью данной бакалаврской работы является разработка программной системы, позволяющей распознавать лица в видеопотоках в режиме реального времени с использованием метода Виолы-Джонса и локальных бинарных шаблонов.

Для достижения поставленной цели, необходимо решить следующие задачи:

- Проанализировать существующие подходы для распознавания лиц;
- Выявить классификацию алгоритмов распознавания;
- Провести анализ существующих на рынке систем распознавания, выявить их достоинства и недостатки;
- Проанализировать основные инструментальные средства для разработки и выбрать оптимальные из них;
- Спроектировать и программно реализовать систему распознавания.

Результатом бакалаврской работы является разработанная система распознавания лиц в видеопотоках для пропускной системы института, с помощью которой возможно повысить безопасность в институте.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На сегодняшний день большинство предприятий уже начали использовать в своей охранной системе - СКУД (системы контроля и управления доступом). Ведь благодаря таким биометрическим системам идентификации можно значительно повысить безопасность предприятия и его сотрудников.

Ранее, для распознавания людей, на проходных предприятия устанавливали электронные турникеты со считывателями карт или отпечатков пальцев, но сегодня, в связи с бурным развитием биометрических технологий, компании все чаще переходят на другие методы распознавания, более точные и удобные. Например, распознавание лиц по видеопотоку в режиме реального времени.

В связи с таким стремительным ростом потребностей предприятий в биометрических системах распознавания, аналитики TrendForce ожидают, что рост интереса к технологиям распознавания лиц в ближайшие годы увеличится еще больше. По предварительным данным, рынок видеоаналитики к 2019 году достигнет 450 млн. долларов. Основная область применения технологии все так же будет связана с системами безопасности СКУД и системами мониторинга, но область их использования с каждым годом будет расширяться[1].

В связи с этим была определена цель данной выпускной квалификационной работы - разработка программной системы контроля доступа в видеопотоках на основе алгоритмов распознавания лиц.

Для достижения поставленной цели требуется:

- Выявить классификацию алгоритмов распознавания;
- Провести анализ существующих на рынке систем распознавания;
- Определить основные инструментальные средства для разработки;
- Спроектировать систему распознавания.

1 Системы распознавания лиц

Процессом распознавания лиц принято называть набор различных задач, служащих для идентификации человека по цифровому изображению или видеофрагменту. В общем виде этот процесс выглядит следующим образом: после того, как система получила изображение с камеры, с помощью алгоритмов определяются границы лица (этап обнаружения). Далее следует этап распознавания, на котором лицо трансформируется (изменяется его яркость, оно выравнивается, масштабируется, и т.п.) и приводится к некоторому заданному виду. После чего, происходит вычисление признаков и непосредственно сравнение их с заложенными в базу данных эталонами. Этот заключительный этап сравнения называется идентификация или верификация, в зависимости от системы.

Верификация: сравнение образцов по схеме «1:1». Для определения личности система сравнивает биометрический образец с одним биометрическим шаблоном, хранящимся в базе данных, и дает ответ на вопрос «Является ли этот человек тем, с чьим шаблоном его сравнивали?».

Идентификация: сравнение образцов по схеме «1:N». Для определения личности система сравнивает биометрический образец со всеми шаблонами лиц, хранящимися в базе данных, и дает ответ на вопрос «кто это?».

На рисунке 1 изображен общий алгоритм распознавания лиц по изображению.

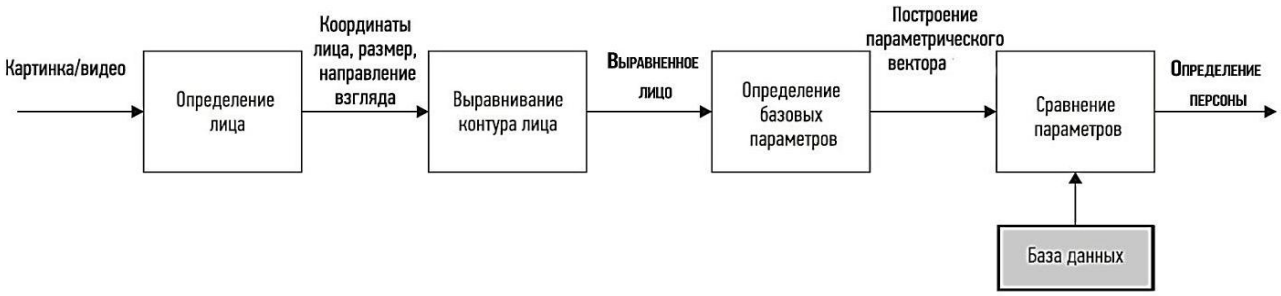

Рисунок 1 – Общий алгоритм распознавания лиц

1.1 Обзор методов обнаружения лиц на цифровых изображениях

После того, как изображение в виде цифровых данных с камер передается на компьютер – оно обрабатывается с помощью специального алгоритма, который определяет расположение области лица по его основным чертам (глазам, рту, бровям, носу и т. д.). Таких методов обнаружения лиц существует много и большинство из них представляют собой комбинацию других методов. Но все их можно разбить на две категории: методы на основе знаний, которые основываются на опыте человека и методы обнаружения лица по внешним признакам (методы при которых необходимо провести этап обучения системы, путём обработки тестовых изображений). Классификация этих методов обнаружения приведена на рисунке 2.

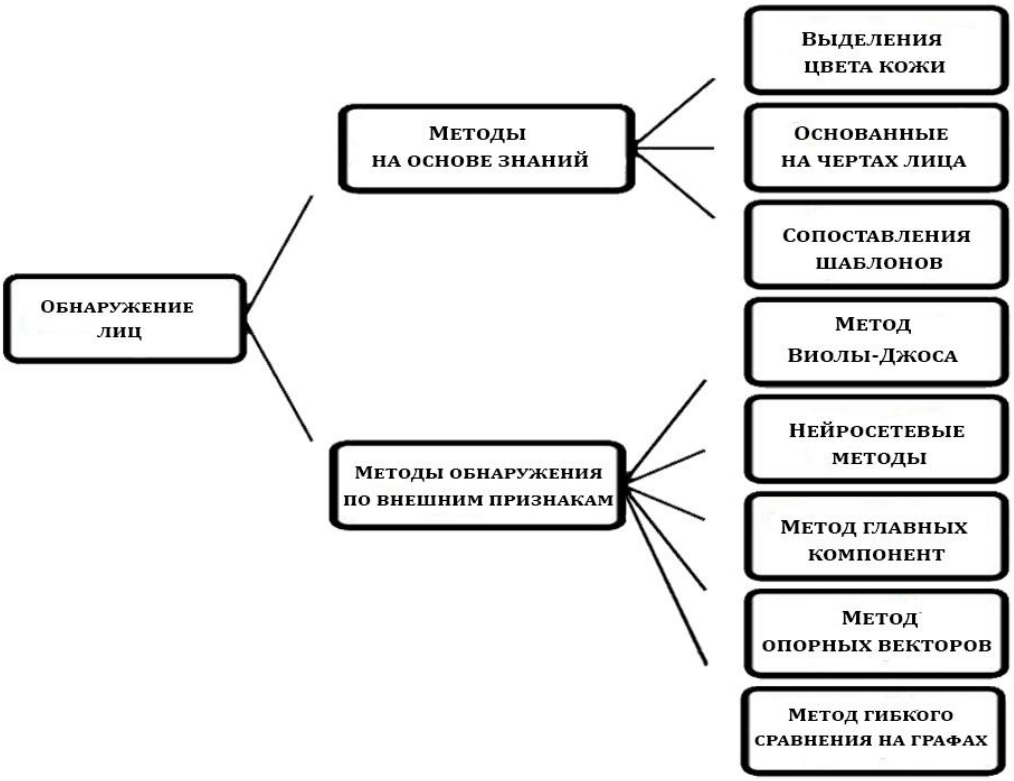

Рисунок 2 – Классификация методов обнаружения лиц

Методы, основанные на знаниях, используют информацию о лице, его чертах, форме, текстуре или цвете кожи. В этих методах выделяется некий набор правил (свойств и особенностей лица), которым должен отвечать фрагмент кадра, для того чтобы считаться человеческим лицом. Определить такой набор правил довольно легко. Все правила это формализованные знания, которыми руководствуется человек, когда определяет, лицо перед ним или не лицо.

Например, основные правила: области глаз, носа и рта отличаются по яркости относительно остальной части лица; глаза на лице всегда располагаются симметрично относительно друг друга. Опираясь на эти и другие похожие свойства, строятся алгоритмы, которые в ходе выполнения проверяют наличие правил на изображении.

К этой же группе методов относят более общий метод - метод сравнения с шаблоном. В этом методе, с помощью описания свойств отдельных областей лица и их заданному взаимному расположению определяется стандарт лица (шаблон), с которым в дальнейшем сравнивают исходное изображение.

Методы на основе знаний получили довольно широкое распространение и имеют неплохие показатели, однако они показывают хорошие результаты только на изображениях с хорошим расширением, без шумов и с несложным фоном. На кадрах с видеопотока или камер установленных в общественных местах, где возможны различные ракурсы и повороты лиц, а также меняющееся освещение и большое количество мешающих объектов на заднем плане, существует большая вероятность возникновения ошибок.

Методы обнаружения лиц по внешним признакам подходят к проблеме с другой стороны, они не пытаются в явном виде формализовать происходящие в человеческом мозге процессы, а наоборот стараются выявить закономерности и свойства изображения лица неявно, применяя методы математической статистики и машинного обучения. Методы этой группы лишены выше отмеченных недостатков и поэтому стали чаще применятся в системах видеонаблюдения. Обнаружение лиц в таких методах осуществляется перебором всех прямоугольных

фрагментов изображения с целью определения, к какому классу относится изображение: к классу содержащих лицо или к классу изображений без лица.

За счет такого большого объёма работы методы обладают избыточностью и большой вычислительной сложностью. Чтобы уменьшить количество вычислений и ускорить процесс отыскания лиц, авторы применяют различные дополнительные методы для сокращения количества рассматриваемых фрагментов.

Несколько наиболее актуальных и заслуживающих внимания методов обнаружения лиц рассмотрены ниже:

Метод Виолы-Джоса (*Viola–Jones object detection*) . Метод был предложен Паулом Виолой и Майклом Джонсом в 2001 году и стал первым методом, демонстрирующим высокие результаты при обработке изображений в реальном времени. У метода имеется множество реализаций, в том числе в составе библиотеки компьютерного зрения OpenCV (функция *cvHaarDetectObjects()*) [2]. Подробно данный метод рассмотрен в разделе 2.

Преимущества данного метода:

- Высокая скорость работы (за счет использования каскадного классификатора);
- Высокий точность обнаружения повернутых лиц на угол до 30 градусов (если угол больше, эффективность данного метода сильно снижается);

Недостатки:

- Длительное время обучения. Алгоритму необходимо проанализировать большое количество тестовых изображений;
- При обнаружении, на положение лица имеются ограничения.

Метод гибкого сравнения на графах (Elastic graph matching). Метод относится к 2D моделированию. Его суть заключается в сопоставлении графов, которые описывают лица (лицо представляется в виде сетки с индивидуальным расположением вершин и ребер).

Процедура распознавания происходит следующим образом - эталонный граф, характеризующий основной параметр распознания, остается без изменения,

в то время как другие деформируются под влиянием структура лица с привязкой к основным антропометрическим точкам: расстояние между глазами, ушами, линия носа, ширина губ и т.п. Чем больше этих точек используется, тем точнее будет система распознавания, но и существенно увеличится время на обработку одного объекта.

Недостатки метода:

- Сложность алгоритма распознания приводит к необходимости использования значительных вычислительных мощностей;
- Сложная процедура введения новых шаблонов в базу данных;
- Быстродействие аналитической системы обратно пропорционально размерам баз данных.

Скрытые Марковские модели (СММ). Метод основан на статистическом сравнении объекта с базой шаблонов. Скрытые Марковские модели используют статистические свойства сигналов и учитывают их пространственные характеристики. Элементы модели: начальная вероятность состояний, множество наблюдаемых состояний, множество скрытых состояний, матрица переходных вероятностей. Каждому элементу соответствует своя Марковская модель. Во время распознавания человека, проверяются все сгенерированные Марковские модели и ищется наибольшая из наблюдаемых вероятность того, что последовательность наблюдений для объекта сгенерирована соответствующей моделью.

Недостатки:

- Низкая скорость срабатывания;
- Низкая различающая способность и не оптимальный алгоритм обучения;
- Система может оптимизировать только время обработки данных и отклика на собственную модель, но не может минимизировать время перебора других моделей.

Метод главных компонент (PCA). Главной целью PCA является уменьшение пространства признаков без значимой потери информации и так, чтобы оно как можно лучше описывало «типичные» образы, принадлежащие множеству лиц. В задаче распознавания лиц его используют главным образом для того, чтобы представить лицо как вектор малой размерности, который затем сравнивается с эталонными векторами из базы.

Набор собственных векторов, который был получен один раз на обучающей выборке, используется для кодирования остальных изображений лиц, которые можно представить взвешенной комбинацией собственных векторов. При использовании ограниченного количества собственных векторов можно получить сжатую аппроксимацию входному изображению лица, которую впоследствии можно хранить в БД, как вектор коэффициентов, который служит одновременно ключом поиска в БД.

PCA хорошо зарекомендовал себя в приложениях. Однако, тогда, когда на изображении лица имеются значительные изменения в выражении лица или освещённости, эффективность метода значительно падает. Это происходит из-за того, что метод главных компонент выбирает подпространство с целью максимальной аппроксимации входного набора данных, а не с целью выполнения дискриминации между классами лиц.

Метод опорных векторов (Support Vector Machines, SVM) - это набор схожих алгоритмов обучения с учителем, использующихся для задач классификации и регрессионного анализа.[4] Суть метода опорных векторов заключается в нахождении гиперплоскости в признаковом пространстве, отделяющий класс изображений лиц от изображений "не-лиц". При этом из всех возможных гиперплоскостей, разделяющих два класса, необходимо выбрать такую гиперплоскость, расстояние до которой от каждого класса максимально.

Преимущества данного метода:

- Высокая устойчивость к переобучению;
- Высокая скорость работы по сравнению с нейронными сетями;

– Возможность уменьшения чувствительности к шуму за счет снижения точности.

Недостатки:

– Точность работы метода уступает многим методам.

Нейросетевые методы. Довольно распространенные методы, которые включают в себя около десятка различных алгоритмов. Основной особенностью таких сетей является их обучаемость на наборе готовых примеров заранее занесенных в базу данных. Во время обучения нейронных сетей, сеть автоматически извлекает ключевые признаки и строит взаимосвязь между ними. После этого, для того чтобы распознать раннее неизвестный объект, обученная нейронная сеть применяет полученный опыт. Нейросетевые методы показывают одни из самых лучших результатов в области распознавания, но считаются наиболее сложными для реализации.

Преимущества данного метода:

– Высокая точность обнаружения при правильной настройке параметров сети.

Недостатки:

– Сложная процедура внесения изменений (внесение любого изменения требует переобучения сети);

– Трудно формализовать архитектуру сети (количество нейронов, слоев, характер связей)

– Высокая вычислительная сложность.

1.2 Предварительная обработка изображений

Для повышения эффективности распознавания и качества выделения лиц, в системах распознавания проводят этап предобработки входных изображений.

После того, как система находит лицо на кадре и определяет положение головы, размер, позу и его ключевые черты, оно, как правило, подвергается

нормализации: кодируется, масштабируется, преобразуется до горизонтального положения линии, соединяющей центры глаз. Также, при предобработке лица, применяются разнообразные фильтры для снижения уровня шума (медианный, гауссовский и пр.).

Методы предварительной обработки довольно разнообразны и зависят от задач исследований. Наиболее часто встречаются из них:

Нелинейные фильтры – фильтры, которые используют для удаления импульсного шума (отдельных точек с максимальной (белой) или минимальной (черной) яркостью) на изображении. Действия таких фильтров заключается в определении позиции каждого импульса и замене их значениями фиксированной или случайной величины.

Фильтр Гаусса – фильтр размытия изображения, который используют, если на изображении существуют мелкие детали, которые не требуют отделения от фона и их можно размыть.

Медианные фильтры – фильтры, использующиеся для сохранения перепадов яркости контуров и подавления импульсных шумов. Медианные фильтры достаточно хорошо работают в таких случаях, при которых плотность шума невелика.

1.3 Методы распознавания лиц

Другая важная часть систем автоматического распознавания лиц это непосредственно сами алгоритмы распознавания человека. На сегодняшний день таких алгоритмов много, и каждый из них имеет свою специфику, свою скорость работы и свою надежность распознавания.

Алгоритмы распознавания делятся на две категории, в зависимости от применяемой технологии распознавания – двумерные, в которых распознавание происходит по геометрии лица (2D-технологии) и трехмерные, в которых распознавание происходит по строению черепа (3D-технологии).

Системы 2D-распознавания работают с «плоскими», двухмерными изображениями и распознают лицо, анализируя его текстуру и участки лица с высокой контрастностью, поэтому при нарушении освещения или положения лица его распознавание сильно затрудняется.

Системы 3D-распознавания же более устойчивы к таким изменениям, т. к. при создании модели лица, в них учитываются особенности строения черепа. Но тем не менее, хоть системы 3 и имеют такой большой плюс, из-за отсутствия возможности обслуживать большое количество пользователей в режиме идентификации и из-за невысокой скорости, 3D-технологии пока не получили широкого применения. Также, 3D-технологии распознавания требуют больших вычислительных ресурсов, и стоимость оборудования для таких систем намного выше, чем у 2D [5].

На сегодняшний день в системах видеонаблюдения часто используется несколько эффективных алгоритмов распознавания, все они основаны либо на значениях пикселей, либо на характерных точках. Ниже представлено краткое описание каждой из этих подгрупп.

Методы, основанные на значениях пикселей, используют для распознавания обнаруженных лиц цвет или яркость. Простейшими подобными методами являются:

Eigenfaces – алгоритм, предложенный в 1991 году Мэтью Тѐ рком и Алексом Пентландом и получивший широкую известность в качестве первого успешного метода распознавания лиц.

Основной идеей алгоритма Eigenfaces является представление отличительных характеристик изображения лица, с помощью двумерных изображений в градациях серого. Для идентификации личности алгоритм сравнивает полученное лицо с кадра, с уже зарегистрированным шаблоном в базе, и определяет коэффициент различия. Этот коэффициент различия между шаблонами и показывает степень схожести.

Алгоритм EigenFaces показывает высокие результаты при использовании в хорошо освещенных помещениях и когда есть возможность сканирования лица в фас, что касается изменений условий использования данного алгоритма, показатели его эффективности резко снижаются.

Fisherfaces – потомок Eigenface, обеспечивающий более высокую точность распознавания при изменениях освещения или выражения лица. В отличие от метода eigenfaces, в основе *Fisherfaces* лежит линейный дискриминантный анализ LDA, а именно линейный дискриминант Фишера.

Действие алгоритма *Fisherfaces* основано на поиске проекции данных, при которой классы изображений лиц максимально разделимы. Данное отличие с Eigenface позволяет решить проблему высокой чувствительности к изменениям освещения.

Локальные бинарные шаблоны (Local Binary Pattern, LBP) – простой в реализации и эффективный метод распознавания лиц. При распознавании методом LBP, каждому пикселю изображения при помощи функции присваивается значение яркости, описывающее его окрестность. Полученное изображение разделяется на области, для каждой из которых рассчитывается гистограмма. Далее гистограммы конкатенируются и сравниваются при помощи методов машинного обучения. В классическом варианте используется метод ближайшего соседа.

В отличие от EigenFaces, алгоритм устойчив к монотонным изменениям освещения, что делает его подходящим для распознавания лиц в системах обработки в реальном времени.

Методы, основанные на характерных точках в отличие от предыдущих, не оценивают яркости пикселей, а используют координаты характерных точек на изображении. Такими характерными точками могут быть, например, центры глаз, положение носа, линия бровей, рта и т.д. К данному классу методов относятся активные модели внешнего вида и активные модели формы.

Активные модели внешнего вида (Active Appearance Models, AAM) — это статистические модели изображений, которые путем разного рода деформаций могут быть подогнаны под реальное изображение. Данный тип моделей в двумерном варианте был предложен Тимом Кутсом и Крисом Тейлором в 1998 году. [6] Активная модель внешнего вида содержит два типа параметров: параметры, связанные с формой (параметры формы), и параметры, связанные со статистической моделью пикселей изображения или текстурой (параметры внешнего вида). Перед использованием модель должна быть обучена на множестве заранее размеченных изображений. Разметка изображений производится вручную.

Активные модели формы (Active Shape Models, ASM) учитывают статистические связи во взаимном расположении антропометрических точек. На каждом изображении выборки эксперт размечает расположение антропометрических точек. Для того чтобы привести координаты на всех изображениях к единой системе обычно выполняется т.н. обобщенный прокрустов анализ, в результате которого все точки приводятся к одному масштабу и центрируются. Далее для всего набора образов вычисляется средняя форма и матрица ковариации. На основе матрицы ковариации вычисляются собственные вектора, которые затем сортируются в порядке убывания соответствующих им собственных значений. Локализации ASM модели на новом, не входящем в обучающую выборку изображении осуществляется в процессе решения оптимизационной задачи.

1.4 Оценка эффективности систем распознавания

Между специализированными и любительскими системами распознавания существует большая разница. Любительские - выглядят красивее, дороже, и к ним предъявляются более низкие требования. Также, за счет того, что в случае допущенных ошибок данными системами, последствия обычно незначительны, их

стоимость гораздо ниже стоимости профессиональных систем, требования к которым предъявляются серьезнее.

Для примера можно взять технологию распознавания лиц на фотографиях в социальных сетях, и профессиональную систему, предназначенную для поиска разыскиваемых людей. Первая система работает с фотографиями, на которых люди обычно смотрят в объектив, а вторая - с людьми, снятыми в общественном месте, которые не смотрят в камеры видеонаблюдения, а иногда даже специально от них скрываются. Также стоит заметить, что если первая система допустит ошибку, то ничего ужасного не произойдет, а вот если система поиска разыскиваемых людей будет допускать много ошибок, то это серьезный недостаток, так как любой из пропущенных людей может оказаться вором, убийцей или даже террористом.

Поэтому при выборе системы важно правильно определить не только цель, но и критерии для оценки эффективности этой системы. А это значит определить приспособленность системы к работе в определенных условиях и различные пороги допустимых ошибок.

В настоящее время для оценки эффективности систем распознавания выделяют 2 основных параметра. В биометрии их называют Коэффициентом ложного доступа (FAR — False Acceptance Rate) и Коэффициентом ложного отказа (False Rejection Rate — FRR).

Коэффициент ложного отказа показывает вероятность того, как часто лица, бывают ошибочно отвергнуты, или другими словами - система не нашла в базе «своего» человека. Чем этот коэффициент меньше, тем выше точность распознавания.

Коэффициент ложного допуска – это наоборот, вероятность того, как часто система по ошибке признает подлинность.

Существуют еще один коэффициент, позволяющий сравнивать биометрические системы - коэффициент EER (равный уровень ошибок). Это коэффициент, при котором обе ошибки (ошибка приёма и ошибка отклонения)

эквивалентны. Чем ниже коэффициент EER, тем выше точность биометрической системы

На рисунке 3 показаны взаимосвязи характеристик FAR, FRR и EER.

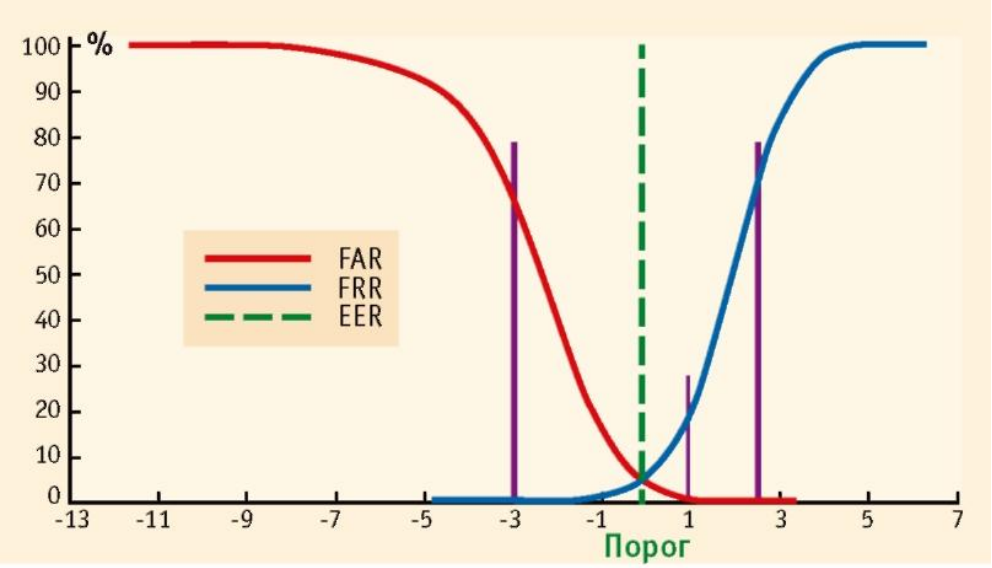

Рисунок 3 – Характеристики биометрических систем

Для сравнения систем и визуализации зависимости между ошибками используют график ROC (Receiver Operating Characteristic).

Кривая ROC определяет, насколько образец должен быть близок к шаблону, чтобы система могла считать это совпадением. При регулировании порогов ошибок, стоит не забывать, что величины FAR и FRR взаимообратные и если уменьшить, то будет меньше ложных несовпадений, но больше ложных приёмов.

1.5 Современные системы распознавания лиц

С каждым годом рост интереса к технологиям распознавания лиц заметно увеличивается. На сегодняшний день уже существует несколько десятков систем распознавания, которые применяются в различных сферах человеческой жизни. Лидерами среди компаний разрабатывающих такие системы считаются:

корпорация NEC (Япония), Cognitec Systems GMBH (Германия), «VisionLabs» (Россия) «FACE++»и Neurotechnology (Литва).

1.5.1 Система «FaceVACS » компании «Cognitec Systems»

«FaceVACS-VideoScan» – простое в использовании, настраиваемое программное обеспечение распознавания лиц по видеопотоку в реальном времени, предлагаемое компанией «Cognitec Systems».

Система «FaceVACS-VideoScan» состоит из нескольких системных компонентов: видеосервера управляющего видеопотоками; сервера видеосканирования координирующего все компоненты системы и выполняющего основные биометрические операции; вычислительного узла, используемого для распределения вычислительной нагрузки; пользовательского интерфейса; диспетчера сигналов получающего уведомления о событиях и обслуживающего мобильные устройства; операционной базы данных; и комплекта интеграторов.

На сегодняшний день технология FaceVACS использует алгоритм распознавания лиц B10T9. Этот алгоритм, устойчив к изменениям мимики, поворотам лица (на ±15º) частичному его закрытию, использованию очков и изменению освещения[7].

Кроме того, система FaceVACS имеет следующие особенности:

- возможность одновременного отслеживания нескольких лиц;
- сравнение лиц происходит в реальном времени;
- возможность отображения и отправки статистики о потоках;
- поддержка интерактивной регистрации из неподвижного;
- применение C++ API и Web Services API.

1.5.2 Система «NEC’s Face Recognition» компании «NEC»

«NEC’s Face Recognition» – одна из передовых систем распознавания лиц, разработанная японской компанией «NEC», позволяющая идентифицировать людей по кадрам многолетней давности и даже, если человек находится в очках или гримасничает. Все распознанные лица хранятся в базе данных, поэтому в случае необходимости можно поднять всю историю видеорегистрации и просмотреть дату и время любого сохраненного изображения.

Технология NEC превосходит множество других систем распознавания своей точностью и скоростью. Она имеет хорошие показатели производительности в различных ситуациях, в том числе при работе с видео низкого качества и сильно сжатыми изображениями. NEC анализирует индивидуальные особенности лица (размер, форму зрачков, линии носа и рта), их взаимное расположение, и находит потом по этой информации подходящего человека в базе данных.

Система включает в себя несколько модулей, реализующих следующие алгоритмы:

1. Используется метод обобщенного соответствия (GMFD), который обеспечивает высокую скорость детектирования и высокую точность распознавания лица. Метод GMFD основан на нейронных сетях и осуществляет предварительный поиск пар глаз;

2. Алгоритм PSM (Perturbation Space Method), позволяет эффективно справляться со сложностями связанными с расположением лица в кадре (лицо под наклоном или некоторым углом).

3. Метод ARBM (Adaptive Regional Blend Matching), который уменьшает воздействие небольших изменений на лице (например, изменения выражения лица, наличие очков, головного убора) на точность распознавания[8].

Система распознавания лиц NeoFace обладает следующими особенностями:

- Возможность наблюдения и контроля в реальном времени;
- Идентификация на основе индивидуальных черт лица;

- Множественное распознавание;
- Возможность поиска событий по базе данных;
- Ведение журнала изображений лиц;
- Устойчивость к повороту лица на ±15º и наклону головы до 45º в любом направлении от фронтального положения;
- "Drag and Drop" управление;
- Масштабируемый и неограниченный размер Базы Данных;
- Независимое распознавание направления взгляда и характеристик лица (очки, борода и выражение лица).

1.5.3 Система «LUNA SDK» компании «VisionLabs»

«LUNA SDK» – специализированное ПО, сопровождения и распознавания лиц людей на цифровых фотографиях или в видеопотоке от компании «VisionLabs». Движок имеет одни из лучших в мире показателей полноты и точности распознавания в реальных условиях.

Процесс распознавания лиц в LUNA SDK работает на основе глубинных нейронных сетей и состоит из нескольких ключевых этапов: определения положения и размеров всех лиц (детекции); определения расположения характерных черт лица и трансформации его в стандартизированную форму (выравнивание); извлечение дескрипторов (числовых векторов, которые суммируют характерные признаки лица); сравнения лиц с базой изображений и предотвращения подмены лиц.

Технические характеристики и особенности LUNA SDK:

- Полностью разработана на C++;
- Возможность наблюдения и контроля в реальном времени;
- Возможность работать в режиме обширной многопоточности;
- Использование алгоритма распознавания лиц на основе глубинных нейронных сетей;

- Оптимизированное управление памятью;
- Компактные дескрипторы, которые обеспечивают вычислительную эффективность;
- Поддержка различных платформ, включая Windows и Linux.

Дескрипторы LUNA обучены на миллионах лиц из различных источников и обеспечивают высокую точность в различных условиях, например, в банках, в системах видеонаблюдения и в социальных сетях. Ниже приведена оценка точности системы LUNA на реальном примере из клиентской базы данных[9].

1.5.4 Система «VeriLook SDK» компании «Neurotechnology»

«VeriLook SDK» – технология идентификации лиц, разработанная компанией «Neurotechnology». Представляет собой систему обнаружения лиц с возможностью одновременного множественного распознавания людей присутствующих в кадре и быстрой идентификации лиц (находит до 100000 лиц за секунду). VeriLook SDK доступна в виде комплекта для разработки ПО и поддерживает широкий выбор устройств на Windows Linux, Mac OS X, iOS и Android[10].

Алгоритм VeriLook реализует локализацию лица с использованием алгоритмов обработки цифровых изображений основанных на глубоких нейронных сетях.

Основными преимуществами системы VeriLook являются отсутствие необходимости контакта со средствами сканирования и быстрое внедрение функций биометрической идентификации в прикладные системы заказчика, а также существует ряд других преимуществ:

- *Одновременная обработка нескольких лиц;*
- *Гендерная классификация.* По желанию, пол может быть определен для каждого человека на изображении;

– *Живое распознавание лица.* VeriLook определяет, является лицо в видеопотоке «живым» или фотографией;

– *Распознавание эмоций.* VeriLook анализирует шесть основных эмоций: гнев, отвращение, страх, счастье, печаль и удивление;

– *Атрибуты лица*. VeriLook может быть настроено для обнаружения определенных атрибутов во время извлечения лица - улыбки, открытого рта, закрытых глаз, очков, бороды или усов;

– *Определение качества изображения лица.* Порог качества может использоваться во время регистрации лица, чтобы гарантировать, что в базе данных будут храниться только шаблоны лучшего качества.

– *Несколько образцов одного и того же лица.* Эти образцы могут быть зачислены из разных источников и в разное время, что позволяет улучшить качество сопоставления;

– *Идентификационная способность.* Функции VeriLook могут использоваться в сопоставлении 1-к-1 (проверка), а также в режиме 1-ко-многим (идентификация);

– *Малый шаблон лица.* Шаблон лица может быть размером от 4 килобайт;

– *Особенности режима обобщения.* Этот режим генерирует коллекцию обобщенных функций лица из нескольких изображений одного и того же объекта;

– Алгоритм VeriLook способен сопоставлять грани, которые были захвачены в инфракрасном спектре.

2 Разработка системы распознавания лиц в видеопотоках

2.1 Этапы обработка кадров видеопотока

Как было отмечено ранее, обработка кадров видеопотока состоит из двух основных этапов. На первом этапе происходит обнаружение лиц в кадре, а на втором – непосредственно само распознавание обнаруженных лиц.

При разработке системы, в данной работе был использован метод обнаружения лиц Виолы-Джонса, а для распознавания – метод ближайшего соседа с использованием гистограмм центрально-симметричных локальных бинарных шаблонов.

Распознавание лиц в системе производится на основе поиска минимального расстояния между гистограммой входного изображения лица и гистограмм, хранящихся базе.

Также стоит сразу отметить, что производительность выбранных алгоритмов существенно зависит от различных факторов, например: освещения, положения лица, заднего фона и т.д. Поэтому, для обеспечения корректной работы системы, необходимо создать благоприятные условия для ее использования:

- Фронтальное, либо близкое к нему положение лица;
- Лицо, не перекрываемое другими объектами;
- Нейтральное выражение лица;
- Тестовая выборка должна сниматься в одинаковых условиях освещения.

Помимо основных этапов обнаружения и распознавания, в разрабатываемой системе существуют промежуточные этапы обработки найденных лиц: фильтр Гаусса, которой после обнаружения лиц помогает снизить влияние шумов, а также маска значимых областей, которая позволяет убрать влияние угловых областей изображения, содержащих задний план.

В результате обобщенный алгоритм обработки кадров разрабатываемой содержит следующие этапы: обнаружение лиц, обработка найденных лиц с

помощью фильтра гаусса, LBP трансформация найденных лиц с последующим применением маски значимых областей, расчет гистограмм найденных лиц, классификация лиц методом ближайшего соседа по гистограммам. Общий алгоритм разрабатываемой системы представлен на рисунке 4.

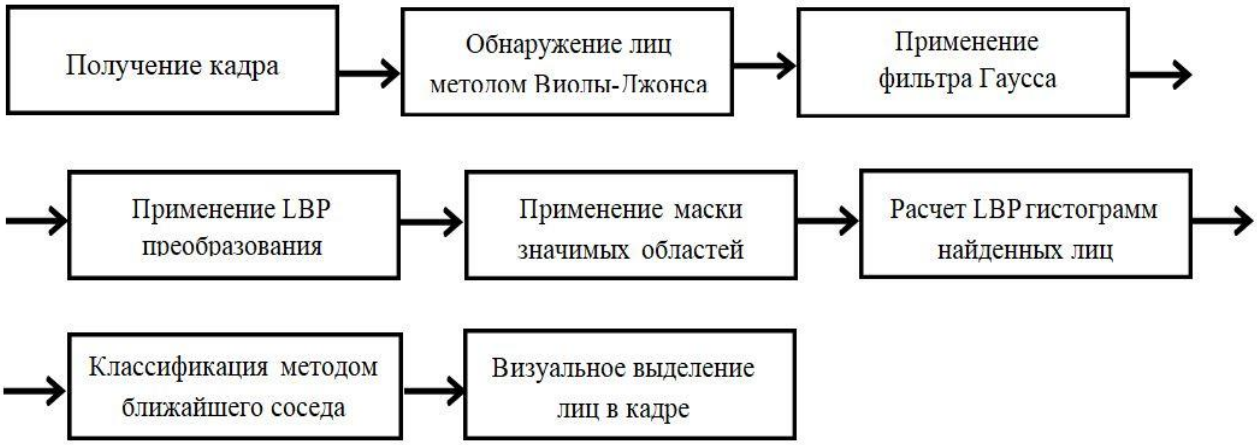

Рисунок 4 – Обобщенный алгоритм обработки кадров видеопотока

Также, по составленному обобщенному алгоритму обработки видеопотока, был составлен необходимый функционал разрабатываемой системы:

- Вывод видеопотока с подключенной к компьютеру камеры в реальном времени или возможность загрузки видео из файла;
- Вывод информации о принадлежности распознаваемого лица к определенному классу;
- Графическое отображение гистограммы и LBP представление отслеживаемого лица;
- Возможность обучения и добавления классов лиц с использованием камеры через интерфейс приложения;
- Возможность настройки параметров работы алгоритмов.

2.1.1 Обнаружение лиц методом Виолы-Джонса

Для разработки системы распознавания, был выбран метод обнаружения лиц Виолы-Джонса. Этот метод был разработан в 2001 году, но благодаря своей высокой скорости, а также крайне низкой вероятности ложного обнаружения лица до сих пор является одним из основных методов поиска объектов на изображении.

Основные принципы, на которых основана работа данного метода:

- Представление изображения в интегральном виде;
- Поиск лиц с помощью признаков Хаара;
- Каскадная классификация;
- Обучение системы распознавания объектов на основе метода «AdaBoost»;

Для повышения эффективности работы с данными, в методе Виолы-Джонса применяется метод интегрального представления, который позволяет быстро рассчитывать сумму яркости пикселей произвольного прямоугольника на заданном кадре. Такое представление изображения представляет собой матрицу одинаковую по размерам с исходным изображением, каждый элемент которой хранит в себе сумму интенсивностей всех пикселей, находящихся над ним и слева от него плюс его собственный вес.

В расширенном методе виолы-Джонса, который используется в библиотеке компьютерного зрения OpenCV и применяется в разрабатываемой системе, используются дополнительные признаки Хаара. Каскады Хаара представляют собой прямоугольные области, которые составлены из нескольких соседних прямоугольных областей, отмеченных как светлая или темная, примеры таких дополнительных признаков представлены на рисунке 5.

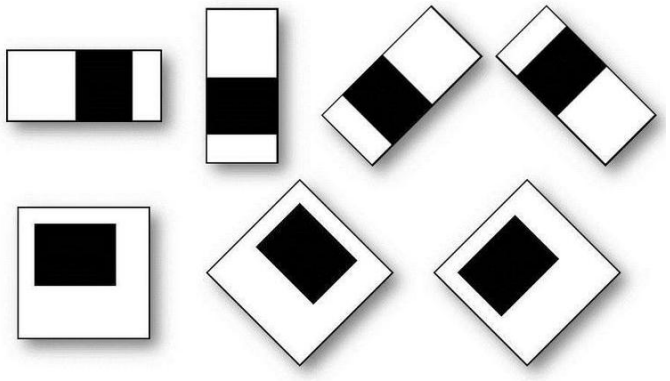

Рисунок 5 – Дополнительные признаки Хаара

Вычисляемым значением таких дополнительных признаков является разность сумм значений яркостей точек закрываемых светлой частью признака и точек, закрываемых темной частью признака.

Поиск лиц происходит при помощи так называемого сканирующего окна, алгоритм сканирования которого выглядит так:

- есть исследуемое изображение, выбрано окно сканирования, выбраны используемые признаки;
- далее окно сканирования начинает последовательно двигаться по изображению с шагом в 1 ячейку окна (допустим, размер самого окна есть 24*24 ячейки);
- при сканировании изображения в каждом окне вычисляется приблизительно 200 000 вариантов расположения признаков, за счет изменения масштаба признаков и их положения в окне сканирования;
- сканирование производится последовательно для различных масштабов;
- масштабируется не само изображение, а сканирующее окно (изменяется размер ячейки);
- все найденные признаки передаются классификатору, который определяет по их значениям, является ли область изображения, соответствующая окну, лицом или нет.

Поскольку для описания объекта с достаточной точностью необходимо большее число признаков Хаара, они не очень подходят для обучения или классификации. В связи с этим, для ускорения процесса обнаружения, в методе Виолы-Джонса используется каскадный классификатор, который позволяет ускорить обнаружение лиц, фокусируя работу на наиболее интересных областях изображения[11][12].

Таким образом при малых вычислительных затратах можно на ранних этапах распознавания отбросить изображения, с большой долей вероятности не содержащие искомый объект (в данном случае лицо). Каждый уровень каскада обучается при помощи алгоритма AdaBoost.

2.1.2 Фильтр Гаусса

С целью устранения шумов на изображениях лиц был применен фильтр Гаусса. Фильтр Гаусса – это фильтр размытия изображения, который использует нормальное распределение (также называемое Гауссовым распределением) для вычисления преобразования, применяемого к каждому пикселю изображения.

Размытие по Гауссу позволяет избавиться от нежелательных шумов на изображениях, и сводит к минимуму их влияние при классификации лиц.

2.1.3 LBP преобразование

LBP оператор впервые был предложен в 1996 году для классификации текстур [13]. Однако позже нашел применение и для распознавания лиц. Суть оператора заключается в применении к пикселям изображения порогового преобразования, в котором значение яркости обрабатываемого пикселя сравнивается со значениями яркостей пикселей его окрестности. Результат сравнения каждого пикселя окрестности с обрабатываемым пикселем конкатенируется в двоичное число.

После применения LBP оператора, изображение делится на прямоугольные области, для каждой из которых рассчитываются гистограммы, описывающие, насколько часто встречаются в данной области пиксели различных значений яркости.

Полученные гистограммы нормализуются, конкатенируются и используются в дальнейшем в качестве признаков классификации. Пример разбиения изображения на прямоугольные области и формирования гистограмм можно увидеть на рисунке 6.

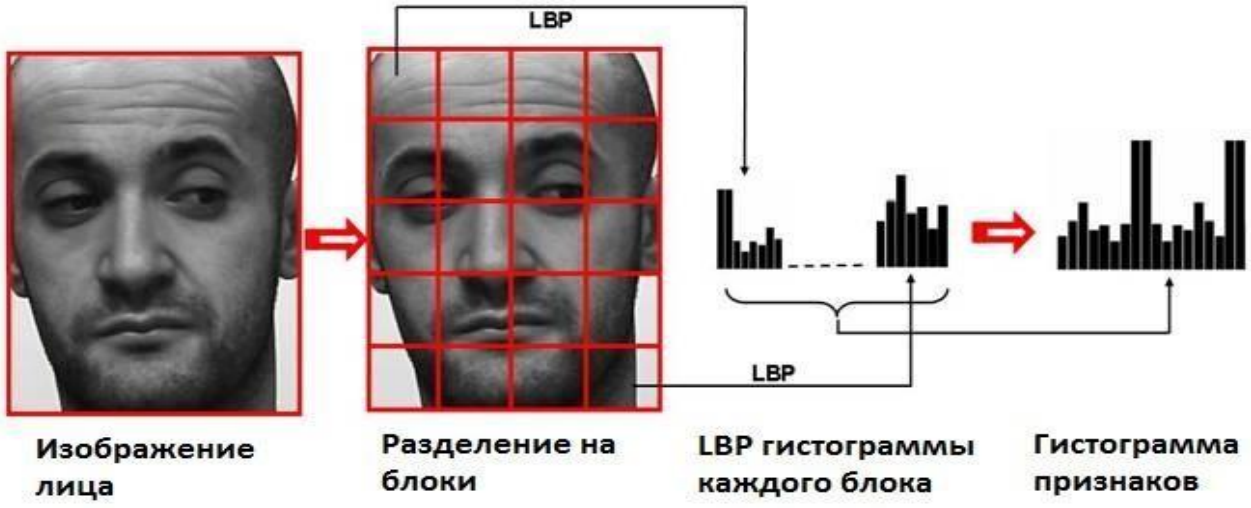

Рисунок 6 – Разбиение изображения на прямоугольные области и формирование гистограммы

В результате получается описание изображения лица на трех уровнях локализации. При этом такое описание не зависит от монотонных изменений освещения.

2.1.4 Маска значимых областей изображения

Изображения лиц, получаемые после процедуры обнаружения, имеют квадратную форму. Однако лицо занимает не все пространство такого

изображения. Поэтому логично было бы исключить влияние на решение классификатора областей изображения, в которых нет лица.

Простым способом решения данной проблемы является применение маски значимых областей изображения. Такая маска представляет собой изображение одного размера с обрабатываемым изображением. Пиксели ненулевой яркости в маске соответствуют значимым областям. В нашем случае значимой областью

При решении задачи классификации с использованием локальных бинарных шаблонов маску значимых областей целесообразно применить после выполнения LBP преобразования и перед расчетом гистограмм. Таким образом, все незначимые пиксели изображения на гистограмме будут сгруппированы в одно значение. Пример применения маски значимых областей к изображению, обработанному LBP оператором, приведен на рисунке 7.

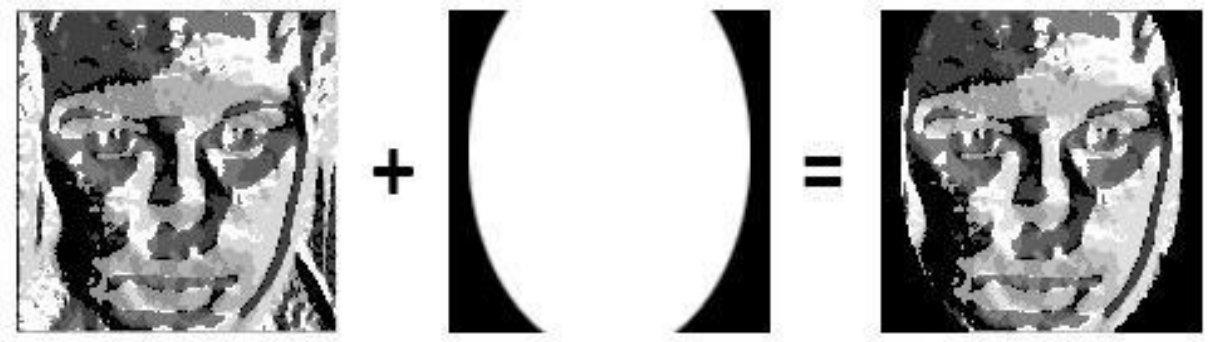

Рисунок 7 – Применение маски значимых областей

Как видно из рисунка, в результате применения данной операции пиксели изображения, которые не должны влиять на результат классификации, принимают нулевое значение яркости.

2.1.5 Метод ближайшего соседа

Метод ближайшего соседа является простым алгоритмом классификации, суть которого заключается в том, что объект относится к тому классу, к элементу

которого он ближе всего находится. Например, на рис. 8 зеленый круг в соответствии с данным алгоритмом должен быть классифицирован как красный треугольник.

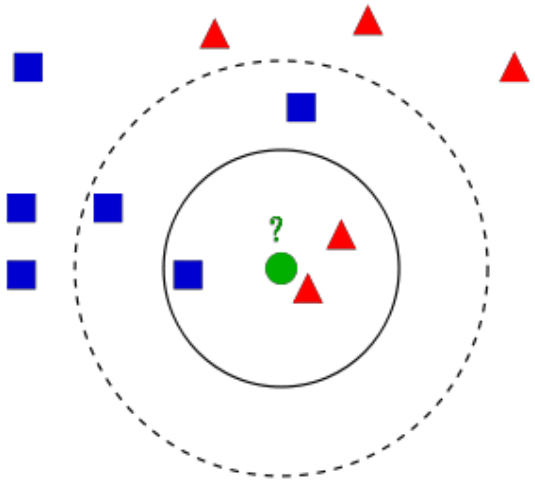

Рисунок 8 – Метод ближайшего соседа

Для улучшения результатов также используют метод, в котором объект относят к тому классу, к которому относится большинство его соседей в окрестности заданного размера. Однако при решении задачи классификации лиц такой подход негативно влияет на работу классификатора.

Данный метод применяют в том случае, когда цена ошибки неправильной классификации является большой, а ошибки данных невелики. Основным недостатком метода ближайшего соседа является его чувствительность к значениям отдельных (возможно ошибочных) данных. Несмотря на это метод показывает высокую эффективность при применении в широком спектре задач классификации[14].

Особого внимания также заслуживает вопрос выбора метрики, определяющей расстояние между гистограммами. Для достижения максимальной точности классификации необходимо выбрать ту метрику, которая наиболее адекватно бы отражала различия между гистограммами изображений разных классов.

2.2 Инструментарий разработки

Разработка системы велась на объектно-ориентированном языке программирования C# в среде разработки Microsoft Visual Studio 2017.

Microsoft Visual Studio — линейка продуктов компании Майкрософт, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств.

В качестве основного был выбран язык C#, так как обладает нужными качествами для реализации, имеет встроенную поддержку обобщений, делегатов и событий, что облегчит реализацию.

C# относится к семье языков с C-подобным синтаксисом, имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML[15].

Немаловажным критерием в пользу выбора данного языка программирования был опыт разработки на нем, полученный за время обучения.

Для разработки системы, также было решено использовать в качестве одной из основных - библиотеку OpenCV. Данная библиотека разработана на C/C++ и имеет обертку для .NET языков – EmguCV, которая и была использована в работе. EmguCV содержит алгоритмы для обработки, реконструкции и очистки изображений, распознания образов, захвата видео, слежения за объектами, калибровки камер и др. Библиотека распространяется по лицензии BSD, а значит, может свободно использоваться в академических и коммерческих целях. [16][17]

2.3 Описание классов

В данном подразделе приведено описание основных классов, функций и методов разработанной программы.

Главный класс приложения MainForm содержит список классов лиц типа FaceClass и список распознанных лиц типа RecognizedFace. Так же он использует в своей работе детектор лиц FaceDetector и LBP преобразователь изображений LBPTransformer. Каждое распознанное лицо RecognizedFace при этом ссылается на соответствующий ему класс лиц FaceClass.

Класс главной формы приложения содержит элементы интерфейса приложения, а также список распознанных и список нераспознанных лиц, которыми будет оперировать система при обработке кадров. Помимо этого данный класс реализует функции расчета гистограмм и графического отображения результатов работы системы. Подробное описание элементов интерфейса и механизмов пользовательского взаимодействия с ними будут описаны в разделе, посвященном разработке интерфейса системы.

RecognizedFace

Класс RecognizedFace описывает лицо распознанное классификатором. Элементы данного класса хранятся в обновляемом списке распознанных лиц, на основе данных которого производится отображение информации на форме.

Атрибуты класса:

private FaceClass _faceClass – класс лиц, к которому принадлежит распознанное лицо.

private Rectangle _rect – прямоугольная область, которая соответствует положению лица в кадре видеопотока.

private Mat _histogram – LBP гистограмма лица.

private Image<Gray, Byte>_face – обрабатываемое в текущий момент изображение лица, взятое из кадра видеопотока.

private Image<Gray, Byte> _lbp – LBP представление изображения_face.

private double _distance – дистанция рассчитываемая между LBP гистограммой лица и LBP гистограммой ближайшего элемента его класса.

Методы класса:

public RecognizedFace(FaceClass fc, Rectangle r, Image<Gray, Byte> f, Image<Gray, Byte> l, Mat h, double d) – конструктор класса. В качестве параметров которого передаются значения, присваиваемые атрибутам создаваемого объекта.

Класс LBPTransformer описывает CS-LBP преобразователь. Данный класс является статическим классом, не имеет атрибутов и содержит только один метод, выполняющий цетрально-симметричное LBP преобразование изображения.

Методы класса:

public static Image<Gray, Byte> CSTransform(Image<Gray, Byte> input) - осуществляет LBP преобразование входного изображения (input) и возвращает преобразованное изображение.

FaceDetector

Класс FaceDetector описывает детектор лиц, использующий в своей работе метод Виолы-Джонса.

Атрибуты класса:

private CascadeClassifier _haar – каскадный классификатор, с помощью которого производится обнаружение лиц.

Методы класса:

public FaceDetector() – конструктор класса. Загружает в виде xml файла в атрибут _haar обученный каскад для распознавания лиц.

public Rectangle[] GetFacesRect(Image<Bgr, Byte> frame, double scaleFactor, int minNeighbors, int sz) – получает на вход изображение (frame), на котором производится поиск лиц, а так же данные для настройки параметров обнаружения, а именно фактор увеличения сканирующего окна (scaleFactor), минимальное количество вложенных обнаружений (minNeighbors) и минимальный размер лиц (sz). Возвращает список прямоугольников, соответствующих положениям лиц на изображении (frame).

Стоит отметить, что минимальное количество вложенных обнаружений напрямую влияет на точность обнаружения лиц. Данный параметр задает необходимое для признания области изображения лицом количество срабатываний

детектора при различных масштабах сканирующего окна, изменяющего свои размеры в соответствии с фактором увеличения scaleFactor, в данной области изображения.

Сам процесс обнаружения лиц осуществляется при помощи функции CascadeClassifier.DetectMultiScale библиотеки Emgu CV, вызываемой для каскадного классификатора _haar. Данная функция представляет собой реализацию каскадной классификации из метода Виолы-Джонса. Ей же и передаются, описанные выше параметры.

FaceClass

Данный класс описывает класс (категорию) лиц. Каждый класс лиц соответствует конкретному распознаваемому человеку.

Атрибуты класса:

private Image<Gray, Byte> _img – изображение лица, использующееся вместе с названием класса для удобства идентификации и различения классов.

private Mat[] _histogram – массив LBP гистограмм различных изображений лица, соответствующего классу. Использование не одной, а нескольких гистограмм, продиктовано особенностями метода ближайшего соседа, поскольку при распознавании программа ищет наиболее похожего представителя класса из всех имеющихся.

MainWindow

Класс MainWindow описывает главное окно приложения.

Атрибуты класса:

private FaceDetector _detector – детектор лиц.

private List<FaceClass> _faces – список классов лиц, хранящихся в памяти программы в текущий момент.

private List<RecognizedFace> _recognizedFaces – список лиц, распознанных в текущем кадре видеопотока.

private List<Rectangle> _notRecognized – список областей текущего кадра видеопотока, содержащих лица, которые классификатор не может отнести к какому-лицо из имеющихся классов.

private Image<Gray, Byte> _mask – изображение маски значимых областей.

private Image<Gray, Byte>[] _currentFaces – изображения лиц, снятые пользователем для формирования нового класса лиц.

private int _currentFacesCount – текущее число изображений лиц для формирования нового класса.

private Rectangle[] _facesRect – список областей текущего кадра видеопотока, содержащих лица.

private bool _capturing – флаг, показывающий производится ли в настоящий момент захват видеопотока.

private Capture _capture – объект, осуществляющийзахват видеопотока.

private String _videoPath – путь к видеофайлу.

private Mat _frame – текущий кадр видеопотока.

private Image<Bgr, Byte> _frameImg – также текущий кадр видеопотока. Необходим для осуществления ряда операций, невозможных с использованием объекта типа Mat.

Методы класса:

public MainForm() – конструктор класса.

public Mat CalcHistogram(Image<Gray, Byte> image) – возвращает гистограмму изображения image.

private void RecognizeFace(Rectangle rect, Image<Gray, Byte> face, Image<Gray, Byte> lbp, Mat histogram, decimal threshold) – метод осуществляющий распознавание лиц и формирования списков распознанных и нераспознанных лиц. В качестве параметров методу передаѐтся изображение лица face, соответствующая ему область кадра rect, LBP- изображение лица lbp и его гистограмма histogram, а также порог распознавания threshold. На основе этих

данных и списка классов лиц выполняется классификация лица методом ближайшего соседа.

После определения класса лица происходит сравнение расстояния до класса с пороговым значением, в результате чего лицо либо помещается в список распознанных лиц, либо в список нераспознанных лиц.

private void ProcessFaces() – данный метод осуществляет обработку обнаруженных в кадре видеопотока лиц, в соответствии с алгоритмом, блок- схема которого представлена на рисунке 9.

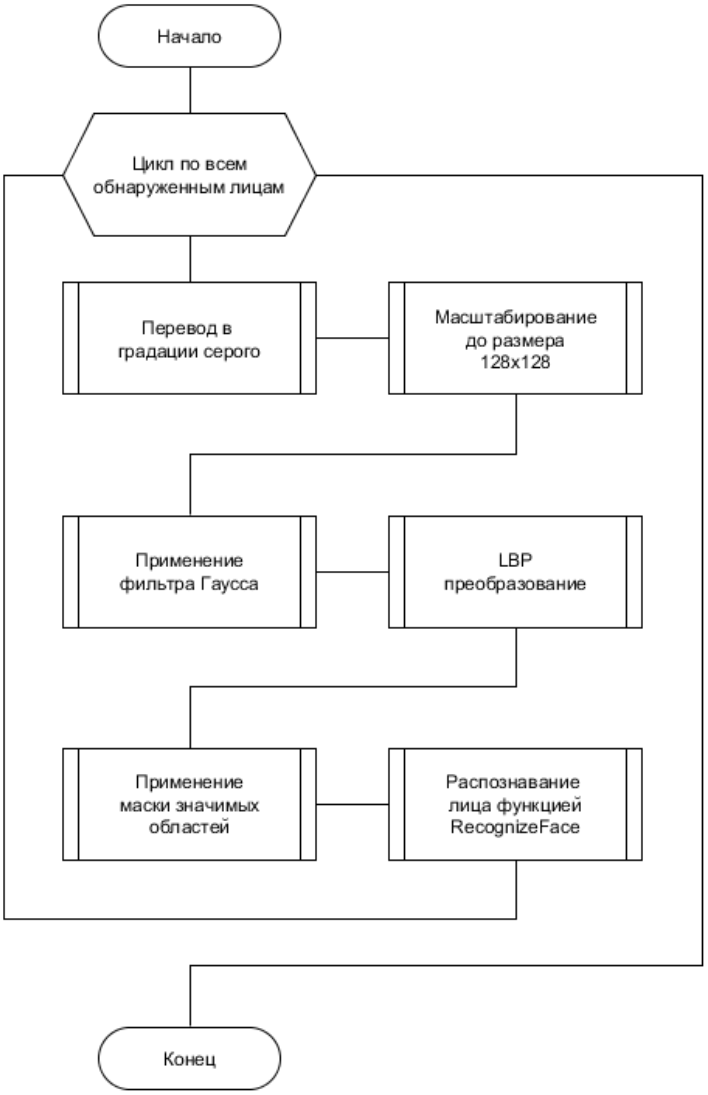

Рисунок 9 – Блок-схема алгоритма обработки обнаруженных лиц

private void ProcessFrame(object sender, EventArgs arg) – функция обработки кадра видеопотока.

private void DrawDetected() - данная функция выполняет визуальное выделение лиц в кадрах видеопотока и вывод информации о выбранном пользователе лице на форму в соответствии с алгоритмом, представленным на рисунке 10.

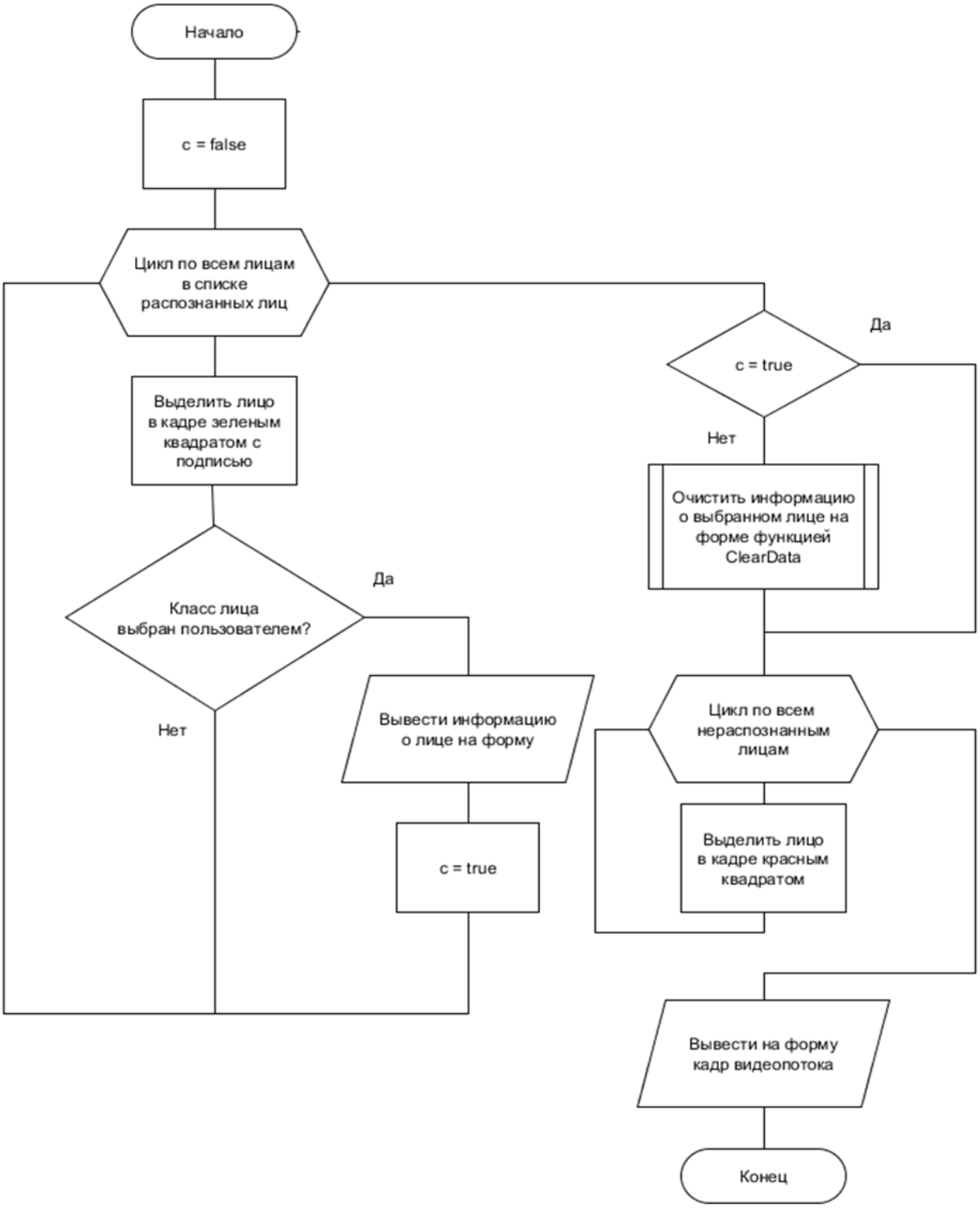

Рисунок 10 – Блок-схема алгоритма вывода информации о лицах на форму

private void ClearData() – метод удаляет с формы информацию о выбранном пользователем лице. Используется каждый раз при обновлении кадра видеопотока перед загрузкой новой информации о лицах.

private void DrawHistogram(Mat histogram) – метод рисует на специальном элементе формы гистограмму histogram.

private void openFaceClass(object sender, CancelEventArgs e) – метод загрузки в приложение класса лица из внешнего файла.

private void addFaceButton_Click(object sender, EventArgs e) – метод добавляет имеющееся в настоящий момент в кадре лицо в список изображений, используемый для формирования нового класса лиц.

private void addFaceButton_Click(object sender, EventArgs e) – метод добавляет имеющееся в настоящий момент в кадре лицо в список изображений, используемый для формирования нового класса лиц.

private void addClassButton_Click(object sender, EventArgs e) – метод добавляет к списку классов лиц новый класс.

private void faceClassesListBox_SelectedIndexChanged(object sender, EventArgs e) – метод, срабатывающий при смене выбранного класса лиц и выводящий на форму изображение нового выбранного класса.

private void removeFaceButton_Click(object sender, EventArgs e) – метод, удаляющий последнее изображение из списка лиц, используемых для создания нового класса.

private void videoButton_Click(object sender, EventArgs e) – метод, запускающий считывание кадров из видеопотока.

private void saveFaceClass(object sender, CancelEventArgs e) – метод, выполняющий сохранение выбранного класса лиц в файл.

private void saveClassButton_Click(object sender, EventArgs e) – метод, вызываемый при нажатии кнопки. Открывает диалоговое окно для сохранения файла.

private void openClassButton_Click(object sender, EventArgs e) – метод, Открывает диалоговое окно для открытия файла.

private void removeClassButton_Click(object sender, EventArgs e) – метод, удаляющий выбранный класс лиц из списка.

private void openVideoButton_Click(object sender, EventArgs e) – метод, вызываемый при нажатии кнопки. Открывает диалоговое окно выбора видеофайла.

private void openFileDialog2_FileOk(object sender, CancelEventArgs e) – сохраняет путь к выбранному видеофайлу в атрибут _videoPath.

Полностью код программы представлен в приложении А.

2.4 Описание интерфейса системы

При разработке интерфейса было реализовано одно главное окно, которое содержит в себе активные элементы для настройки параметров работы приложения, и области вывода данных о распознаваемых лицах.

Интерфейс приложения содержит несколько областей: область вывода видеопотока с камеры или с файла, область настройки параметров работы приложения, область работы с классами лиц и область вывода информации о выбранном пользователем лице. Общий вид главной формы приложения представлен на рисунке 11.

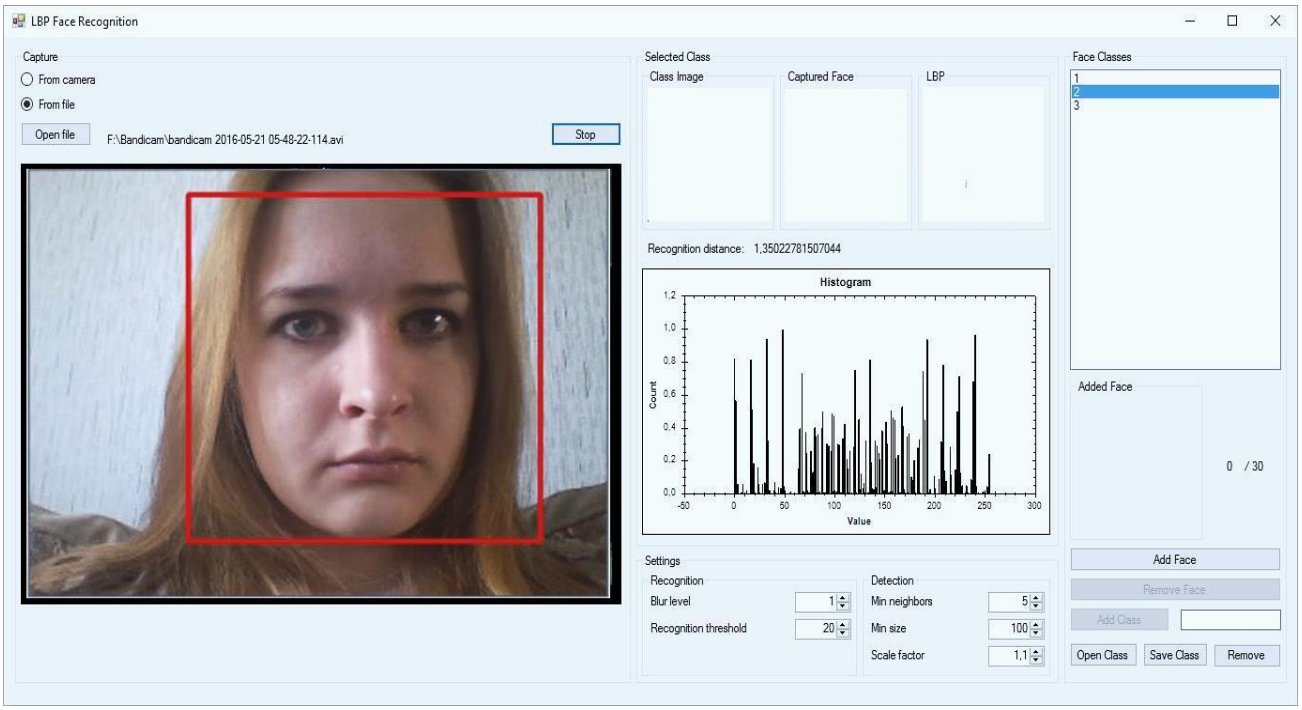

Рисунок 11 – Интерфейс приложения

Рассмотрим подробнее различные области интерфейса разработанной системы.

Область вывода видеопотока предназначена для отображения обработанных кадров. Помимо этого данная область включает кнопки настройки источника видеопотока, кнопку вызова диалогового окна открытия видеофайла, и кнопку, управляющую началом и остановкой захвата видеопотока.

В качестве источника видеопотока может использоваться подключенная к компьютеру веб-камера либо видеофайл. Для выбора видеофайла-источника необходимо нажать кнопку Open File. После выбора файла в появившемся диалоговом окне и нажатия кнопки ОК, путь к выбранному файлу отобразится на форме. Запуск/остановка обработки и вывода кадров из выбранного источника осуществляется нажатием кнопки Start/Stop.

Область вывода видеопотока представлена на рисунке 12.

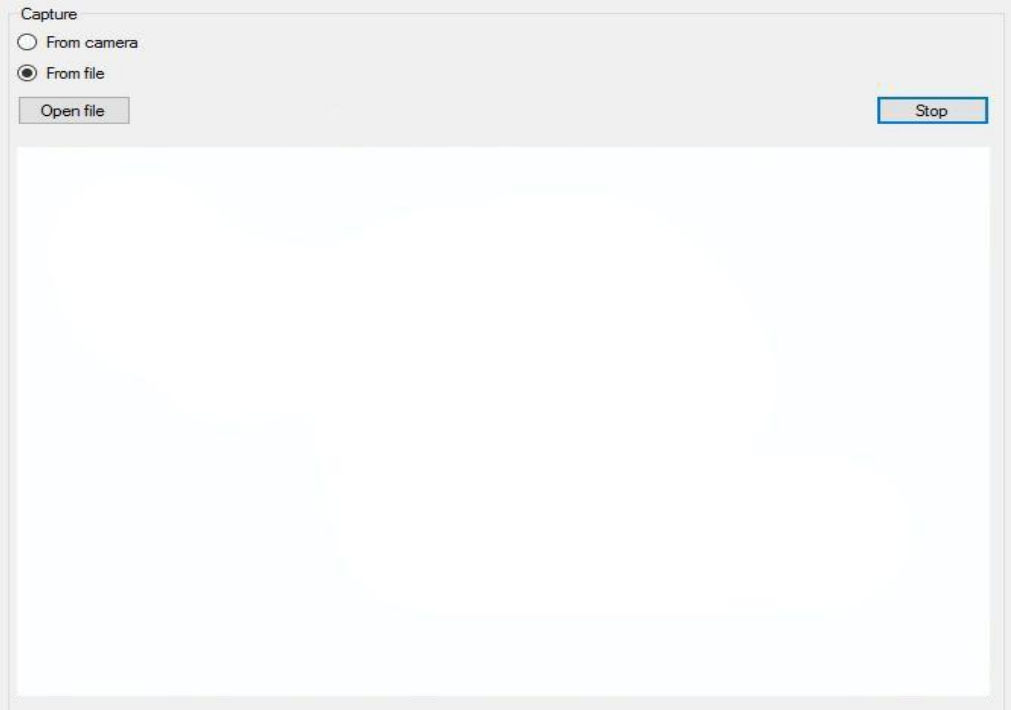

Рисунок 12 − Область вывода видеопотока

Область вывода информации о выбранном пользователем лице предназначена для отображения данных распознаваемых приложением лиц. Источником данных для отображения является выбранный пользователем в списке Face Classes класс лиц и распознанное лицо, соответствующее этому классу.

В данной области формы выводятся изображение выбранного класса, прямоугольная область текущего кадра, соответствующая лицу этого класса, а также LBP преобразованное изображение лица. Помимо этого отображается текущее расстояние между гистограммой распознанного лица и гистограммой его класса и графическое представление гистограммы LBP изображения лица.

Область вывода информации о выбранном пользователем лице представлена на рисунке 13.

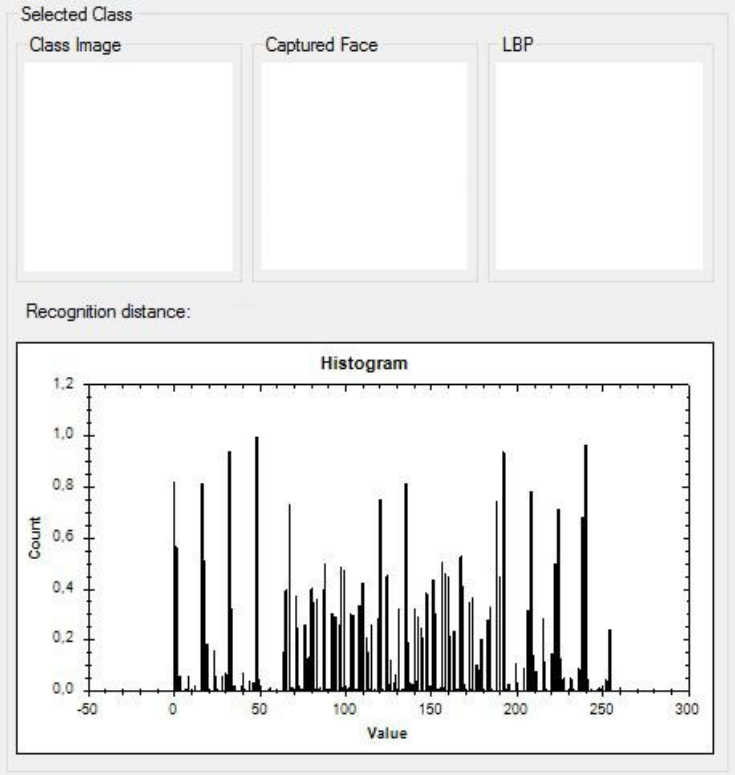

Рисунок 13 – Область вывода информации о выбранном лице

Область настроек работы приложения включает в себя две группы настроек. Первая группа - это настройки работы распознавания лиц. К ним относится уровень размытия по Гауссу и порог распознавания. Порог распознавания – максимально допустимое для принятия решения о принадлежности лица классу расстояние между LBP гистограммой лица и гистограммой класса.

Вторая группа настроек данной области содержит настройки детектора лиц. Значения данных настроек передаются методу GetFacesRect. Область настроек работы приложения представлена на рисунке 14.

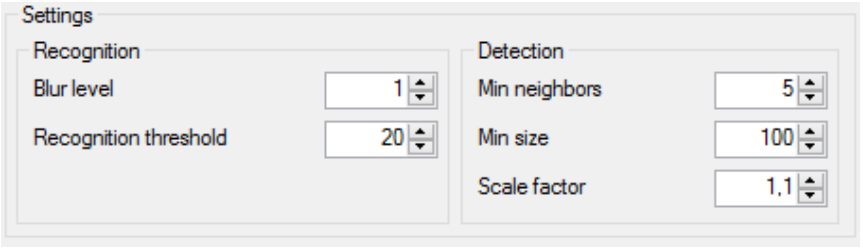

Рисунок 14 – Область настроек работы приложения

Область работы с классами лиц предназначена для работы со списком классов лиц, а также формирования новых классов. В верхней части области находится список загруженных в настоящий момент в программу классов лиц. Пользователь может выбирать нужный ему класс нажатием левой клавиши мыши, и информация о выбранном классе и лицах, ему соответствующих, будет отображаться в области вывода информации.

Ниже расположены элементы интерфейса для формирования нового класса лиц. По нажатию кнопки Add Face в набор изображений для формирования класса добавится изображение лица, присутствующее в настоящий момент в кадре видеопотока. При помощи кнопки Remove Face можно удалить последнее изображение из набора. Как только необходимый набор изображений (до 30 включительно) будет сформирован, необходимо ввести имя класса в текстовое поле и нажать кнопку Add Class для создания на основе набора изображений нового класса лиц. Созданный класс сразу же отобразится в списке классов и будет использоваться при распознавании лиц. Помимо этого в данной области так же присутствуют кнопки для открытия имеющегося класса лиц из файла, сохранения выбранного класса в файл и удаления выбранного класса из списка. Область работы с классами лиц представлена на рисунке 15.

Рисунок 15 − Областьработы с классами лиц

Также стоит отметить, что для предотвращения ошибок, различные элементы интерфейса системы включаются и выключаются в зависимости от активности видеопотока и наличия в нем изображений лиц.

ЗАКЛЮЧЕНИЕ

В результате данной выпускной квалификационной работы была спроектирована система распознавания лиц в видеопотоках на основе метода Виолы-Джонса и локальных бинарных шаблонов.

Разработанная система может использоваться при решениях различных задач видео аналитики, и, в первую очередь, имеет непосредственное применение в системах контроля доступа и идентификации личности.

Кроме этого, при написании работы был проведен обзор современных систем распознавания, выявлены недостатки и трудности, влияющие на эффективность их работы, изучены методы обработки изображений и произведен анализ современных алгоритмов распознавания.

Таким образом, задачи, поставленные перед началом научно-исследовательской работы, были полностью осуществлены.

А основными направлениями дальнейшего развития разработанной системы можно назвать улучшение работы классификатора лиц. Для чего, целесообразно будет заменить алгоритмы классификации на более совершенные.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Analysis market recognition technology [Электронный ресурс]. // TrendForce. Режим доступа: https://www.trendforce.com/ (дата обращения: 2.05.2018).

2. Метод Виолы — Джонса [Электронный ресурс]. // Wikipedia. – Режим доступа: https://ru.wikipedia.org/wiki/Метод_Виолы_—_Джонса (дата обращения: 5.05.2018).

3. Видеоаналитика и распознавание лиц [Электронный ресурс]. // Wikipedia. – Режим доступа: https://video-praktik.ru/st_videoanalitika.html/ (дата обращения: 2.05.2018).

4. Метод опорных векторов [Электронный ресурс]. // Wikipedia. – Режим доступа: https://ru.wikipedia.org/wiki/Метод_опорных_векторов (дата обращения: 15.05.2018).

5. Актуальность 2D алгоритмов в определенных задачах автоматического распознавания [Электронный ресурс]. // Реноме. – Режим доступа: https://moluch.ru/conf/tech/archive/2/138/ (дата обращения: 19.05.2018).

6. Хабрахабр - крупнейший ресурс для IT-специалистов. [Электронный ресурс]. // Режим доступа – https://habr.com/post/133826/ (дата обращения: 1.05.2018).

7. Системы технической безопасности и охраны «MTI». FaceVACS – VideoScan [Электронный ресурс]. // Режим доступа: http://www.security.mti.ua/products/sistemy-videonabludeniya/Soft-raspoznavanie-liz/Cognitec/152-facevacsvideoscan_/ (дата обращения: 12.05.2018).

8. NEC. Facial Recognition [Электронный ресурс]. // Режим доступа: https://ru.nec.com/solutions/security/technologies/face_recognition.html (дата обращения: 12.05.2018).

9. VisionLabs. LUNA SDK [Электронный ресурс]. // Режим доступа: https://visionlabs.ai/ru/luna-platform-info.html (дата обращения: 13.05.2018).

10. Neurotechnology. VeriLook SDK [Электронный ресурс]. // Режим

доступа: http://www.neurotechnology.com/ (дата обращения: 13.05.2018).

11. Локальные бинарные шаблоны [Электронный ресурс]. // Режим доступа: https://ru.wikipedia.org/wiki/Локальные_бинарные_шаблоны (дата обращения: 13.05.2018).

12. Татаренков Д. А. Анализ методов обнаружения лиц на изображении [Электронный ресурс]. // Молодой ученый. — 2015. — №4. — С. 270-276. — Режим доступа: https://moluch.ru/archive/84/15524/ (дата обращения: 11.05.2018).

13. Броневич А. Н. Лекции по методам машинного обучения [Электронный ресурс]. // Режим доступа: http://window.edu.ru/resource/files/lect_Lepskiy_Bronevich.pdf (дата обращения: 05.05.2018).

14. Методы ближайшего соседа и k-ближайших соседей [Электронный ресурс]. // Режим доступа: http://studbooks.net/2429081/informatika/metody_blizhayshego_soseda_blizhayshih_sosedey (дата обращения: 13.05.2018).

15. Язык программирования C#. Классика Computers Science. 4-е изд. / А. Хейлсбер, М. Торгерсен, С. Вилтамут, П. Голд. СПб.: Питер, 2011, 784 с.

16. About OpenCV [Электронный ресурс]. // Режим доступа: http://opencv.org/about.html (дата обращения: 08.05.2018).

17. EmguCV [Электронный ресурс]. // Режим доступа: http://www.emgu.com/wiki/index.php/Main_Page (дата обращения: 18.05.2018).

18. Разработка мультимедийных приложений с использованием библиотек OpenCV и IPP [Электронный ресурс]/ А.В. Бовырин [и др.].— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 515 с.— Режим доступа: http://www.iprbookshop.ru/39564 (дата обращения: 08.05.2018).

19. СТО 4.2-07-2014 Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности.– Введ. 9.01.2014. – Красноярск: ИПК СФУ, 2014. – 60 с.

ПРИЛОЖЕНИЕ А

Листинг программы

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
//подключение библиотек using System;
using System.Drawing;

//подключение бибилотеки Emgu CV using Emgu.CV;
using Emgu.CV.Structure;
using Emgu.CV.Features2D;
using Emgu.CV.UI;
using Emgu.CV;

namespace WindowsFormsApp3
{
    [Serializable]
    //класс, описывающий категорию лиц
    class FaceClass
    {
        //изображение лица
        private Image<Gray, Byte> _img;

        //LBP гистограмма
        private Mat[] _histogram;

        //имя класса
        private String _name;

        //конструктор класса
        public FaceClass(Image<Gray, Byte> faceimg, Mat[] facehist, String facename)
        {
            _img = faceimg;
            _histogram = facehist;
            _name = facename;
        }

        //далее геттеры и сеттеры
        public Mat[] GetHist()
        {
            return _histogram;
        }

        public String GetName()
        {
            return _name;
        }

        public Image<Gray, Byte> GetImg()
        {
            return _img;
        }
    }
}
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Drawing;

//подключение бибилотеки Emgu CV using Emgu.CV;
using Emgu.CV.Structure;
using Emgu.CV.Features2D;
using Emgu.CV.UI;
using Emgu.CV;

namespace WindowsFormsApp3
{
    class FaceDetector
    {

        //каскадный классификатор
        private CascadeClassifier _haar;

        //конструктор класса
        public FaceDetector()
        {
            //загрузка каскадов хаара для распознавания лиц
            _haar = new CascadeClassifier("haarcascade_frontalface_default.xml");

        }

        //возвращает список прямоугольных областей кадра с лицами
        public Rectangle[] GetFacesRect(Image<Bgr, Byte> frame, double scaleFactor,
                    int minNeighbors, int sz)
        {
            try
            {
                Rectangle[] rect = _haar.DetectMultiScale(frame, scaleFactor, minNeighbors,
new                     Size(sz, sz));
                Rectangle[] rect1 = new Rectangle[rect.Length];

                return rect;
            }

            catch (ArgumentOutOfRangeException)
            {
                return null;
            }
        }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Windows.Forms;
//подключение библиотек using System;
using System.IO;
using System.Runtime.Serialization.Formatters.Binary;

//подключение библиотеки EmguCV
using Emgu.CV;
using Emgu.CV.Structure;
using ZedGraph;

namespace WindowsFormsApp3
```

```
{
    public partial class Form1 : Form
    {
        //детектор лиц
        private FaceDetector _detector;
        //классы лиц
        private List<FaceClass> _faces;
        //распознанные лица
        private List<RecognizedFace> _recognizedFaces;
        //нераспознанные лица
        private List<Rectangle> _notRecognized;
        //маска значимых областей
        private Image<Gray, Byte> _mask;
        //изображения для формирования нового класса
        private Image<Gray, Byte>[] _currentFaces;
        //их количество
        private int _currentFacesCount;
        //обнаруженные лица
        private Rectangle[] _facesRect;
        //флаг активности захвата видеопотока
        private bool _capturing;
        //захват видеопотока
        private VideoCapture _capture = null;
        // путь к видеофайлу
        private String _videoPath = null;
        //кадр видеопотока
        private Mat _frame;
        //кадр видеопотока
        private Image<Bgr, Byte> _frameImg;

        //конструктор класса
        public MainForm()
        {
            InitializeComponent();

            _detector = new FaceDetector();
            _faces = new List<FaceClass>();
            _notRecognized = new List<Rectangle>();
            _recognizedFaces = new List<RecognizedFace>();
            _currentFaces = new Image<Gray, Byte>[1000];
            _currentFacesCount = 0;
            _facesRect = null;
            _mask = new Image<Gray, byte>("mask.png");
            _capturing = false;
            _capture = new VideoCapture();

            zedGraphControl1.GraphPane.Title.Text = "Histogram";
            zedGraphControl1.GraphPane.XAxis.Title.Text = "Value";
            zedGraphControl1.GraphPane.YAxis.Title.Text = "Count";
        }

        //расчет гистограммы
        public Mat CalcHistogram(Image<Gray, Byte> image)
        {
            int dim = 4; //разбиение на блоки
            Mat result = new Mat();

            int width = image.Width / dim;
            int height = image.Height / dim;

            //цикл по всем блокам
            for (int i = 0; i < dim; i++)
            {
```

```
                for (int j = 0; j < dim; j++)
                {
                    image.ROI = new Rectangle(i * width, j * height, width, height);
                    DenseHistogram hist = new DenseHistogram(16, new RangeF(0, 15));
                    Image<Gray, Byte>[] inp = new Image<Gray, Byte>[1];
                    inp[0] = image;

                    //расчет для текущего блока
                    hist.Calculate(inp, true, null);
                    CvInvoke.Normalize(hist, hist);

                    //добавление к итоговой гистограмме
                    result.PushBack(hist);
                }
            }

            image.ROI = Rectangle.Empty;
            return result;
        }

        //распознавание лица
        private void RecognizeFace(Rectangle rect, Image<Gray, Byte> face, Image<Gray, Byte>
lbp, Mat histogram, decimal threshold)
        {
            //минимальное расстояние до класса
            double mindist = double.MaxValue;

            //класс-результат
            FaceClass answer = null;

            //цикл по всем классам лиц
            foreach (FaceClass f in _faces)
            {
                //цикл по всем гистограммам класса
                foreach (Mat h in f.GetHist())
                {
                    double       dist       =       CvInvoke.CompareHist(histogram,       h,
Emgu.CV.CvEnum.HistogramCompMethod.Chisqr);

                    //сравнить расстояние с минимальным
                    if (dist <= mindist)
                    {
                        //обновить минимальное расстояние
                        mindist = dist;
                        //принять результатом данный класс лиц
                        answer = f;

                    }
                }
            }

            //если расстояние меньше порогового обновить список распознанных лиц
            if (mindist < (double)threshold)
            {
                bool contains = false;
                CvInvoke.EqualizeHist(lbp, lbp);

                foreach (RecognizedFace element in _recognizedFaces)
                {
                    if ((element.GetFaceClass().GetName() == answer.GetName()) && (mindist <
element.GetDist()))
                    {
                        _notRecognized.Add(element.GetRect());
```

```
                        element.SetFace(face);
                        element.SetLBP(lbp);
                        element.SetRect(rect);
                        element.SetHist(histogram);
                        element.SetDist(mindist);
                        contains = true;
                    }
                    //иначе добавить лицо в список нераспознанных лиц
                    else  if  ((element.GetFaceClass().GetName()  ==  answer.GetName())  &&
(mindist >= element.GetDist()))
                    {
                        contains = true;
                        _notRecognized.Add(rect);
                    }
                }
                if (!contains)
                {
                    _recognizedFaces.Add(new   RecognizedFace(answer,   rect,   face,   lbp,
histogram, mindist));
                }
            }
            else
            {
                _notRecognized.Add(rect);
            }
        }

        //обработка списка лиц
        private void ProcessFaces()
        {
            //для каждого лица в списке обнаруженных
            foreach (Rectangle rect in _facesRect)
            {
                _frameImg.ROI = rect;
                Image<Gray, Byte> face = _frameImg.Convert<Gray, Byte>();
                _frameImg.ROI = Rectangle.Empty;

                //масштабирование
                face = face.Resize(128, 128, Emgu.CV.CvEnum.Inter.Linear);

                //размытие по Гауссу
                CvInvoke.GaussianBlur(face, face, new Size((int)blurUpDown.Value * 2 - 1,
(int)blurUpDown.Value * 2 - 1), 0);
                //LBP преобразованиеLBPHFaceRecognizer
                Image<Gray, Byte> lbpNoMask = LBPTransformer.CSTransform(face);
                Image<Gray, Byte> lbp = new Image<Gray, byte>(128, 128, new Gray(0));
                CvInvoke.cvCopy(lbpNoMask, lbp, _mask);

                //расчет гистограммы
                Mat histogram = CalcHistogram(lbp);
                //распознавание
                RecognizeFace(rect, face, lbp, histogram, thresholdUpDown.Value);
            }
        }
        //обработка кадра
        private void ProcessFrame(object sender, EventArgs arg)
        {
            //обнуление списков лиц
            _recognizedFaces = new List<RecognizedFace>();
            _notRecognized = new List<Rectangle>();
            _frameImg = new Image<Bgr, Byte>(_capture.Width, _capture.Height);
            _frame = new Mat();
```

```
            //получение кадра видеопотока
            _capture.Retrieve(_frame, 0);
            _frameImg = _frame.ToImage<Bgr, Byte>();

            //обнаружение лиц
            _facesRect = _detector.GetFacesRect(_frameImg, (double)scalefactorUpDown.Value,
(int)neighborsUpDown.Value, (int)minsizeUpDown.Value);

            if (_facesRect.Length == 1)
            {
                addFaceButton.Invoke(new Action(() => { addFaceButton.Enabled = true; }));
            }
            else
            {
                addFaceButton.Invoke(new Action(() => { addFaceButton.Enabled = false; }));
            }

            //обработка списка обнаруженных лиц
            ProcessFaces();
            //вывод данных
            DrawDetected();
        }

        //очистка данных с формы
        private void ClearData()
        {
            zedGraphControl1.GraphPane.CurveList.Clear();
            zedGraphControl1.AxisChange();
            zedGraphControl1.Invalidate();
            capFaceImageBox.Image = null;
            lbpImageBox.Image = null;

            distLabel.Invoke(new Action(() =>
            {
                distLabel.Text = " ";
            }));
        }

        //вывод данных на форму
        private void DrawDetected()
        {
            bool contains = false;

            //цикл по всем распознанным лицам

                foreach (RecognizedFace face in _recognizedFaces)
            {
                //выделить зеленым в кадре
                _frameImg.Draw(face.GetRect(), new Bgr(0, 255, 0), 3);
                //написать в кадре имя класса
                _frameImg.Draw(face.GetFaceClass().GetName(),       face.GetRect().Location,
Emgu.CV.CvEnum.FontFace.HersheyComplex, 1, new Bgr(0, 255, 0));

                //вывод данных на форму
                faceClassesListBox.Invoke(new Action(() =>
                {
                    if                  (face.GetFaceClass().GetName()                  ==
(string)faceClassesListBox.SelectedItem)
                    {
                        capFaceImageBox.Image = face.GetFace();
                        lbpImageBox.Image = face.GetLBP();
                        DrawHistogram(face.GetHist());
```

```
                    distLabel.Text = face.GetDist().ToString();
                    contains = true;
                }
            }));
        }
        if (!contains) ClearData();

        //выделить все нераспознанные лица красным
        foreach (Rectangle rect in _notRecognized)
        {
            _frameImg.Draw(rect, new Bgr(0, 0, 255), 3);
        }
        _frameImg    =    _frameImg.Resize(captureBox.Width,    (int)(_frame.Height    *
(captureBox.Width / (double)_frame.Width)), Emgu.CV.CvEnum.Inter.Linear);
        captureBox.Image = _frameImg.ToBitmap();
    }
    //функция отрисовки гистограммы на форме
    private void DrawHistogram(Mat histogram)
    {
        GraphPane pane = zedGraphControl1.GraphPane;
        pane.CurveList.Clear();
        int dim = histogram.Height;
        Image<Gray, Double> histimg = histogram.ToImage<Gray, Double>();
        double[] hist = new double[dim];
        double[] xvalues = new double[dim];

        for (int i = 0; i < dim; i++)
        {
            xvalues[i] = i;
            hist[i] = histimg.Data[i, 0, 0];
        }

        BarItem bar1 = pane.AddBar("", xvalues, hist, Color.DeepSkyBlue);
        zedGraphControl1.AxisChange();
        zedGraphControl1.Invalidate();
    }

    //открытие класса лиц
    private void OpenFaceClass(object sender, CancelEventArgs e)
    {
        foreach (string name in openFileDialog1.FileNames)
        {
            if (File.Exists(name))
            {
                Stream TestFileStream = File.OpenRead(name);
                BinaryFormatter deserializer = new BinaryFormatter();
                FaceClass face = (FaceClass)deserializer.Deserialize(TestFileStream);
                TestFileStream.Close();
                _faces.Add(face); faceClassesListBox.Items.Add(face.GetName());
            }
        }
    }
    //добавление изображения лица в список для формирования класса
    private void AddFaceButton_Click(object sender, EventArgs e)
    {
        if ((_currentFacesCount < 1000) && (_facesRect.Length != 0))
        {
            for (int qwe = 0; qwe < 1000; qwe++)
            {
                Image<Gray, Byte> fr = _frame.ToImage<Gray, Byte>();
                fr.ROI = _facesRect[0];
                _currentFaces[_currentFacesCount]      =      fr.Resize(128,      128,
Emgu.CV.CvEnum.Inter.Linear);
```

```
                    CvInvoke.GaussianBlur(_currentFaces[_currentFacesCount],
_currentFaces[_currentFacesCount],
                        new Size((int)blurUpDown.Value * 2 - 1, (int)blurUpDown.Value * 2 -
1), 0);
                    imageBox4.Image = _currentFaces[_currentFacesCount];
                    _currentFacesCount++;
                    countLabel.Text = _currentFacesCount.ToString();
                    addClassButton.Enabled = true;
                    removeFaceButton.Enabled = true;
                }
            }
        }
        //добавление нового класса лиц
        private void AddClassButton_Click(object sender, EventArgs e)
        {
            //инициализация
            Image<Gray, Byte>[] img = new Image<Gray, Byte>[_currentFacesCount];
            Image<Gray,     Byte>     face     =     _currentFaces[0].Resize(128,     128,
Emgu.CV.CvEnum.Inter.Linear);
            Mat[] hist = new Mat[_currentFacesCount];

            //обработкавсех лиц списка формирования нового класса и расчет гистограмм
                        for (int i = 0; i < _currentFacesCount; i++)
            {
                Image<Gray, Byte> lbpNoMask = LBPTransformer.CSTransform(_currentFaces[i]);
                img[i] = new Image<Gray, byte>(128, 128, new Gray(0));
                CvInvoke.cvCopy(lbpNoMask, img[i], _mask);
                hist[i] = CalcHistogram(img[i]);
            }

            _faces.Add(new FaceClass(face, hist, classNameTextBox.Text));
            faceClassesListBox.Items.Add(classNameTextBox.Text);
            _currentFaces = new Image<Gray, Byte>[1000];
            _currentFacesCount = 0;
            countLabel.Text = "0";
            addClassButton.Enabled = false;
            removeFaceButton.Enabled = false;
            imageBox4.Image = null;
        }

        //смена выбранного класса лиц
        private void FaceClassesListBox_SelectedIndexChanged(object sender, EventArgs e)
        {
            foreach (FaceClass face in _faces)
            {
                if (face.GetName() == (string)faceClassesListBox.SelectedItem)
                {
                    classImageImageBox.Image = face.GetImg();
                }
            }
            if (faceClassesListBox.SelectedItem != null)
            {
                saveClassButton.Enabled = true;
                removeClassButton.Enabled = true;
            }
            else
            {
                saveClassButton.Enabled = false;
                removeClassButton.Enabled = false;
            }
        }
        //удаление класса лиц из списка
        private void RemoveFaceButton_Click(object sender, EventArgs e)
```

```
{
    _currentFacesCount--;
    countLabel.Text = _currentFacesCount.ToString();

    if (_currentFacesCount != 0)
    {
        imageBox4.Image = _currentFaces[_currentFacesCount - 1];
    }
    else
    {
        removeFaceButton.Enabled = false;
        addClassButton.Enabled = false;
        imageBox4.Image = null;
    }
}
//запуск обработки видеопотока
private void VideoButton_Click(object sender, EventArgs e)
{
    if (_capturing)
    {
        startButton.Text = "Start";
        _capturing = false;
        _capture.Stop();
    }
    else
    {
        try
        {
            startButton.Text = "Stop";
            _capturing = true;

            if (radioButton1.Checked)
            {
                _capture.Dispose();
                _capture = new VideoCapture();
                _capture.ImageGrabbed += ProcessFrame;
                _capture.Start();
            }
            else
            {
                _capture.Dispose();
                _capture = new VideoCapture(_videoPath);
                _capture.ImageGrabbed += ProcessFrame;
                _capture.Start();
            }
        }
        catch (NullReferenceException excpt)
        {
            MessageBox.Show(excpt.Message);
        }
    }
}
//сохранение класса лиц в файл
private void SaveFaceClass(object sender, CancelEventArgs e)
{
    FaceClass tosave = null;
    foreach (FaceClass face in _faces)
    {
        if (face.GetName() == faceClassesListBox.SelectedItem.ToString())
        {
            tosave = face;
        }
    }
```

```
            Stream fileStream = File.Create(saveFileDialog1.FileName);
            BinaryFormatter serializer = new BinaryFormatter();
            serializer.Serialize(fileStream, tosave);
            fileStream.Close();
        }
        //далее обработчики нажатий кнопок формы
        private void SaveClassButton_Click(object sender, EventArgs e)
        {
            saveFileDialog1.ShowDialog();
        }

        private void OpenClassButton_Click(object sender, EventArgs e)
        {
            openFileDialog1.ShowDialog();
        }

        private void RemoveClassButton_Click(object sender, EventArgs e)
        {
            _faces.RemoveAll(f => f.GetName() == (string)faceClassesListBox.SelectedItem);
            faceClassesListBox.Items.Remove(faceClassesListBox.SelectedItem);
        }

        private void OpenVideoButton_Click(object sender, EventArgs e)
        {
            openFileDialog2.ShowDialog();
        }

        private void OpenFileDialog2_FileOk(object sender, CancelEventArgs e)
        {
            _videoPath = openFileDialog2.FileName;
            videoPathLabel.Text = _videoPath;
        }
    }}
```